Walmart
Save money. Live better.

About:

Walmart, founded by Sam Walton in 1962 and incorporated in 1969, is one of the largest multinational retail corporations in the world. Headquartered in Bentonville, Arkansas, Walmart operates a vast chain of hypermarkets, discount department stores, and grocery stores. The company's global presence spans multiple countries, serving over 100 million customers worldwide. Known for its commitment to low prices and wide product range, Walmart has established itself as a leader in the retail industry, employing millions of associates globally.

Business Case Study

Retail Formats:

- Supercenters: Large stores offering a wide range of products, including groceries, clothing, electronics, and household goods.
- Discount Stores: Focus on providing a variety of products at lower prices.
- Neighborhood Markets: Smaller stores primarily focused on groceries, pharmaceuticals, and limited household items.

Global Presence:

- Operations in Multiple Countries: Walmart has a significant presence in the U.S., Canada, Mexico, Central America, South America, Asia, and Europe.
- Brand Names: In different regions, Walmart operates under various brand names, such as Asda in the UK and Flipkart in India.

E-Commerce:

- Strong Online Presence: Walmart.com and other e-commerce platforms play a crucial role in Walmart's business model.
- Technology and Logistics: Continuous investment in technology and logistics to enhance online shopping and delivery services.

Philanthropy and Sustainability:

- Community Support Programs: Involvement in numerous initiatives aimed at community support and disaster relief.
- Sustainability Initiatives: Focus on reducing waste, increasing energy efficiency, and supporting sustainable agriculture.

Financials:

- Fortune Global 500: Consistently ranks among the top companies in the Fortune Global 500.
- Revenue Generation: Known for significant revenue and large-scale employment.

Market Expansion Strategy:

- Regional Preferences: To continue expanding its global footprint, understanding regional preferences is crucial. Tailoring products and services to specific demographics and cultural nuances is essential for sustained growth.
- Competitive Landscape: In a highly competitive retail environment, characterized by the emergence of new players and evolving strategies, Walmart must differentiate itself by offering a compelling mix of products and services. This requires a deep understanding of consumer preferences and shopping patterns across various regions.
- Personalization: With the rise of personalized shopping experiences, leveraging data insights to refine product recommendations and anticipate emerging trends is vital. This enhances customer engagement and retention.
- Risk Mitigation: By using data analytics to forecast consumer demand and gauge the potential success of different products, Walmart can mitigate risks associated with inventory and optimize its supply chain strategy.
- Cultural Relevance: Success in international markets often depends on cultural relevance. Understanding cultural sensitivities, societal trends, and local preferences is crucial for creating product assortments and marketing campaigns that resonate with customers across different regions, fostering loyalty and connection to the brand.

Content:

1. Import the Dataset and Basic Data Analysis
2. Detect Null Values and Outliers
3. Data Exploration
4. Confidence Intervals using the Central Limit Theorem (CLT)
5. Poisson Distribution Analysis
6.Box-Cox Transformation
7. Gaussian (Normal) Distribution Analysis
8. Recommendations and Action Items for Walmart

Data:

The company collected the transactional data of customers who purchased products from the Walmart Stores during Black Friday. The dataset has the following features: Dataset link: Walmart_data.csv

User_ID:	**User ID**
Product_ID:	**Product ID**
Gender:	**Sex of User**
Age:	**Age in bins**
Occupation:	**Occupation(Masked)**
City_Category:	**Category of the City (A,B,C)**
StayInCurrentCityYears:	**Number of years stay in current city**
Marital_Status:	**Marital Status**
ProductCategory:	**Product Category (Masked)**
Purchase:	**Purchase Amount**

1. Import the Dataset and Basic Data Analysis

Input:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.pyplot import figure
from scipy import stats
from scipy.stats import boxcox

data = pd.read_csv('/content/walmart_data.csv')

data.head()
data.describe()
```

Output:

```python
data.head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

	User_ID	Purchase
count	1.502550e+05	150255.000000
mean	1.002930e+06	9308.608991
std	1.687846e+03	4982.877654
min	1.000001e+06	160.000000
25%	1.001451e+06	5848.500000
50%	1.002968e+06	8053.000000
75%	1.004339e+06	12062.000000
max	1.006040e+06	23961.000000

Input:

```python
data.info()
# Change the data type
clos = ['Occupation','Marital_Status','Product_Category']
data[clos] = data[clos].astype('object')
```

Output:

```
[35] data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150255 entries, 0 to 150254
Data columns (total 10 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   User_ID                     150255 non-null  int64
 1   Product_ID                  150255 non-null  object
 2   Gender                      150255 non-null  object
 3   Age                         150255 non-null  object
 4   Occupation                  150255 non-null  object
 5   City_Category               150255 non-null  object
 6   Stay_In_Current_City_Years  150255 non-null  object
 7   Marital_Status              150255 non-null  object
 8   Product_Category            150255 non-null  object
 9   Purchase                    150255 non-null  int64
dtypes: int64(2), object(8)
memory usage: 11.5+ MB
```

Dataset Overview

- Total Entries: 550,068
- Total Columns: 10
- Memory Usage: 42.0+ MB

Column Breakdown

- User_ID: Unique identifier for each user.
- Product_ID: Unique identifier for each product.
- Gender: Gender of the user (categorical).
- Age: Age group of the user (categorical).

- Occupation: Numerical code representing the user's occupation.
- City_Category: Categorical data indicating the user's city (A, B, C).
- Stay_In_Current_City_Years: Number of years the user has stayed in the current city (categorical).
- Marital_Status: Marital status of the user (0 = single, 1 = married).
- Product_Category: Numerical code representing the category of the product.
- Purchase: Purchase amount (numerical).

Potential Insights:

- **Demographic Analysis**:
  - Gender Distribution: Proportion of male vs. female users.
  - Age Distribution: Count of users in each age group.
  - Occupation Distribution: Most common occupations among users.
- **Geographical Insights:**
  - City Category: Distribution of users across different city categories (A, B, C).
  - Stay Duration: Average number of years users have stayed in their current city.
- **Marital Status Analysis:**
  - Purchase Behavior: Comparison of purchase amounts between single and married users.
  - Product Preferences: Popular product categories among single vs. married users.
- **Product Analysis:**
  - **Top Products**: Most frequently purchased products.
  - **Product Category Trends**: Popular product categories based on purchase amounts
- **Purchase Behavior:**
  - **Average Purchase Amount**: Overall and segmented by different demographics (age, gender, city category, etc.).
  - **Purchase Distribution**: Range and distribution of purchase amounts.

Visualization Suggestions:

- **Bar Charts**: For gender distribution, age distribution, and city category distribution.
- **Histograms**: For purchase amount distribution.
- **Box Plots**: To compare purchase amounts across different demographics (age, gender, city category).
- **Heatmaps**: To visualize correlation between numerical variables.
- **Pie Charts**: For marital status distribution and product category preferences.

2. Detect Null Values and Outliers:

Input:

```
data.isnull().sum()
```

Output:

```
User_ID                       0
Product_ID                    0
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category              0
Purchase                      0
dtype: int64
```

Insights:

- There are no missing values in the dataset.
- Purchase amount might have outliers.: the max Purchase amount is 23961 while its
- mean is 9263.96. The mean is sensitive to outliers, but the fact the mean is so small
- compared to the max value indicates the max value is an outlier

2.1. Non-Graphical Analysis: Value counts and unique attributes

Input:

```
categorical_cols =
['Gender','Age','Occupation','Marital_Status','Product_Category','Stay_In_Current_City_Years','City_Category']

# Melt the dataframe
melted_data = data[categorical_cols].melt()

# Compute the proportions
proportions = melted_data.groupby(['variable', 'value']).size() /
len(data)

# Reset index for better readability
proportions = proportions.reset_index(name='proportion')

# Display the result
print(proportions)
```

Output:

	variable	value	proportion
0	Age	0-17	0.027180
1	Age	18-25	0.183288
2	Age	26-35	0.396812
3	Age	36-45	0.201251
4	Age	46-50	0.082154
5	Age	51-55	0.070580
6	Age	55+	0.038734
7	City_Category	A	0.267013
8	City_Category	B	0.422635
9	City_Category	C	0.310352
10	Gender	F	0.243446
11	Gender	M	0.756554
12	Marital_Status	0	0.591202
13	Marital_Status	1	0.408798
14	Occupation	0	0.127710
15	Occupation	1	0.083678
16	Occupation	2	0.047692
17	Occupation	3	0.032877
18	Occupation	4	0.132435
19	Occupation	5	0.022129
20	Occupation	6	0.037396
21	Occupation	7	0.107011
22	Occupation	8	0.002722
23	Occupation	9	0.011367
24	Occupation	10	0.023054
25	Occupation	11	0.021683
26	Occupation	12	0.055219
27	Occupation	13	0.014029
28	Occupation	14	0.050441
29	Occupation	15	0.022009
30	Occupation	16	0.046554
31	Occupation	17	0.072537
32	Occupation	18	0.012432
33	Occupation	19	0.015933
34	Occupation	20	0.061089
35	Product_Category	1	0.256557
36	Product_Category	2	0.043959
37	Product_Category	3	0.036970
38	Product_Category	4	0.021590

Insights:

Age Distribution:

- Approximately 80% of the users fall within the age range of 18 to 50 years.
    - 40% are aged between 26 to 35 years.
    - 18% are aged between 18 to 25 years.
    - 20% are aged between 36 to 45 years.

Gender Distribution:

- A significant majority of the users are male, accounting for 75% of the user base.
- Female users constitute 25% of the total users.

Marital Status:

- The user base is predominantly single, with 60% of users not married.
- Married users make up 40% of the total users.

City Residency:

- 35% of the users have been residing in their current city for 1 year.
- 18% have been residing for 2 years.
- 17% have been residing for 3 years.

Product Categories:

- The dataset includes a total of 20 different product categories, indicating a diverse range of products available to users.

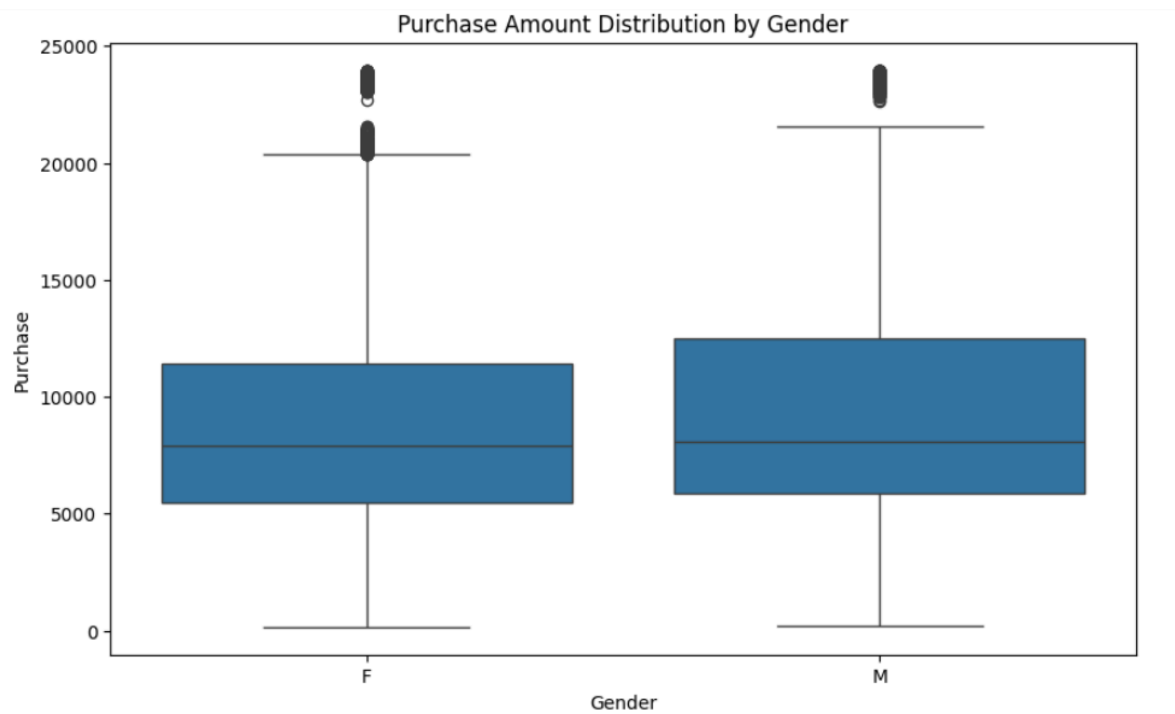
The dataset reveals a user base that is predominantly male, with a significant portion of the users falling within the age range of 18 to 50 years. The majority of users are single and have been residing in their current cities for a relatively short period (1-3 years). The diversity in product categories and occupations highlights a varied consumer base with potentially different purchasing behaviors and needs. These insights can aid in tailoring marketing strategies, product offerings, and customer engagement plans to better meet the needs of different user segments.

2.2. Handling Missing Values and Outliers

Input:

```
# Detect outliers using boxplot
plt.figure(figsize=(10, 6))
sns.boxplot(x='Gender', y='Purchase', data=data)
plt.title('Purchase Amount Distribution by Gender')
plt.show()

# Describe statistics for Purchase amount
print(data['Purchase'].describe())
```

Output:



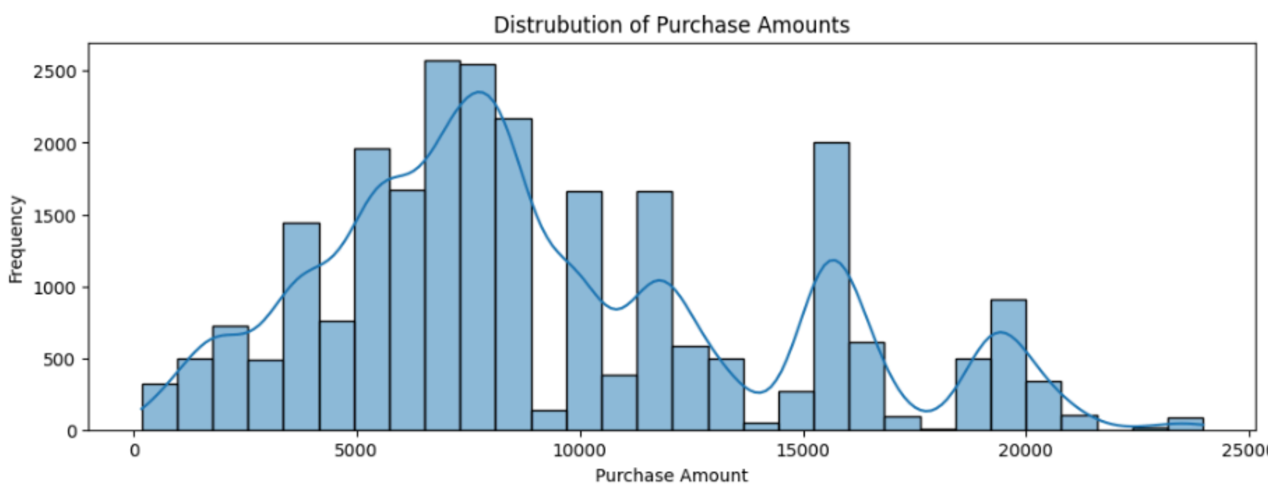

```
count    150255.000000
mean       9308.608991
std        4982.877654
min         160.000000
25%        5848.500000
50%        8053.000000
75%       12062.000000
max       23961.000000
```

2.3 Purchase Analysis:

Input:

```
plt.figure(figsize=(12,4))
sns.distplot(data['Purchase'],kde = True,bins=30)
plt.title('Distribution of Purchase Amounts')
plt.xlabel('Purchase Amount')
plt.ylabel('Frequency')
plt.show()
```

Output:



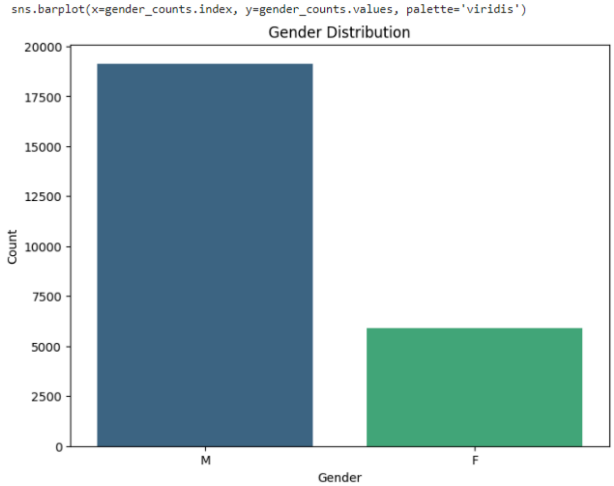
2.4 Gender Base Purchase Analysis:

Input

```
# Gender distribution
gender_counts = data['Gender'].value_counts()
print(gender_counts)

# Average Purchase by Gender
avg_purchase_by_gender = data.groupby('Gender')['Purchase'].mean()
print(avg_purchase_by_gender)

# Bar chart for Gender distribution
plt.figure(figsize=(8, 6))
sns.barplot(x=gender_counts.index, y=gender_counts.values,
palette='viridis')
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```

Output:

```
Gender
M    19139
F     5897
Name: count, dtype: int64
Gender
F    8730.292691
M    9398.277287
Name: Purchase, dtype: float64
<ipython-input-32-9dbcc3d6fcc3>:11: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=gender_counts.index, y=gender_counts.values, palette='viridis')
```



3. Data Exploration:

3.1. Hypothesis Testing and Confidence Intervals

Input:

```
# Hypothesis Testing: Do women spend more on Black Friday than men?
male_purchase = data[data['Gender'] == 'M']['Purchase']
female_purchase = data[data['Gender'] == 'F']['Purchase']

t_stat, p_val = stats.ttest_ind(male_purchase, female_purchase)
print(f"T-statistic: {t_stat}, P-value: {p_val}")

# Confidence Interval Calculation
sample_size = 1000
male_sample = male_purchase.sample(sample_size)
female_sample = female_purchase.sample(sample_size)

male_mean = np.mean(male_sample)
female_mean = np.mean(female_sample)
male_std = np.std(male_sample)
female_std = np.std(female_sample)

confidence_level = 0.95
z_score = stats.norm.ppf((1 + confidence_level) / 2)

male_margin_error = z_score * (male_std / np.sqrt(sample_size))
female_margin_error = z_score * (female_std / np.sqrt(sample_size))

male_confidence_interval = (male_mean - male_margin_error, male_mean +
male_margin_error)
female_confidence_interval = (female_mean - female_margin_error, female_mean +
female_margin_error)

print(f"Male Confidence Interval: {male_confidence_interval}")
print(f"Female Confidence Interval: {female_confidence_interval}")
```

Output:

```
T-statistic: 9.098696612568355, P-value: 9.804470636723891e-20
Male Confidence Interval: (9052.927578734969, 9683.10442126503)
Female Confidence Interval: (8234.810801202126, 8801.495198797875)
```

3.2 Analysis for Marital Status and Age

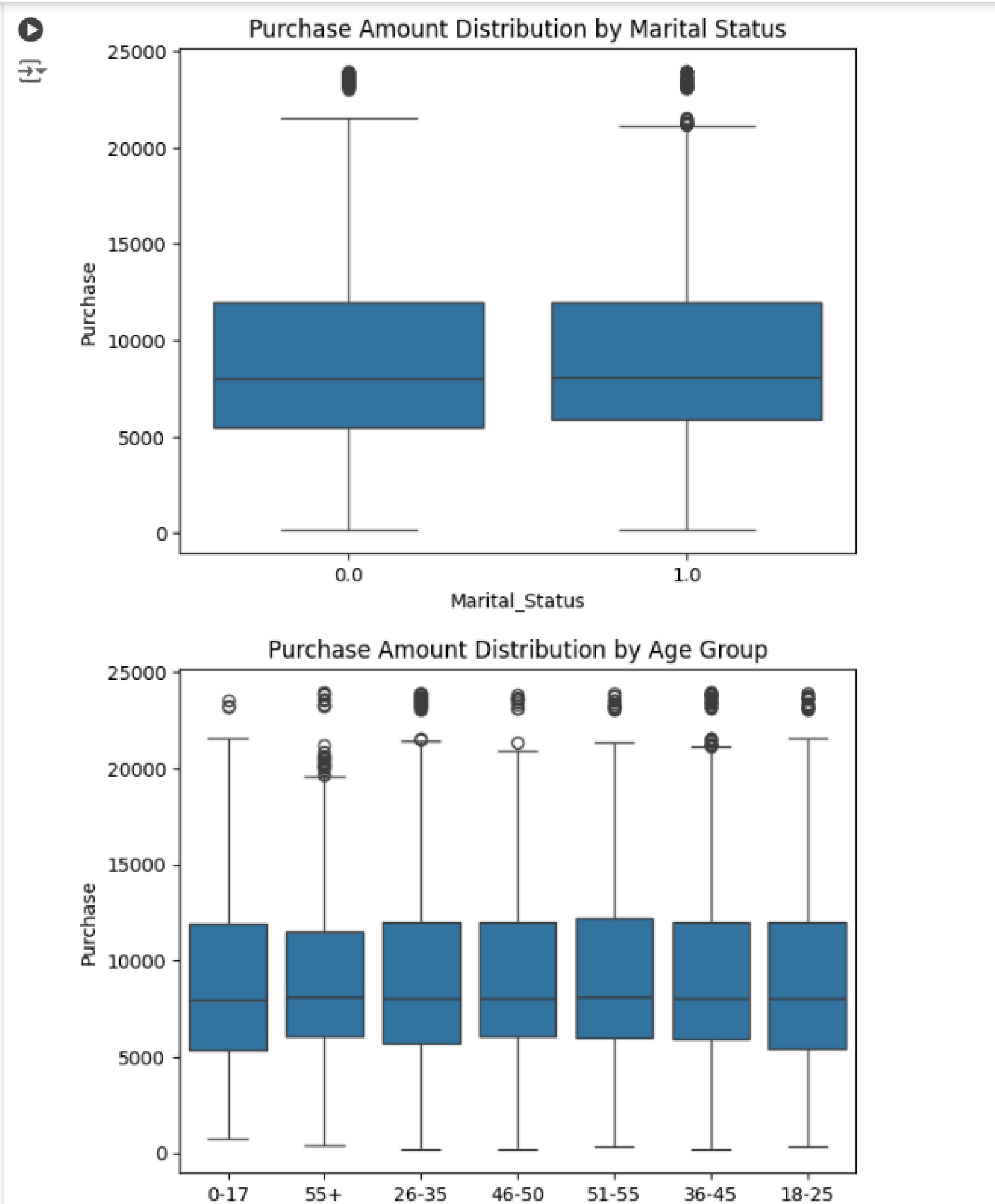
Input:

```
# Marital Status Analysis
marital_avg_purchase = data.groupby('Marital_Status')['Purchase'].mean()
print(marital_avg_purchase)

# Age Group Analysis
age_avg_purchase = data.groupby('Age')['Purchase'].mean()
print(age_avg_purchase)

# Visualization
sns.boxplot(x='Marital_Status', y='Purchase', data=data)
plt.title('Purchase Amount Distribution by Marital Status')
plt.show()

sns.boxplot(x='Age', y='Purchase', data=data)
plt.title('Purchase Amount Distribution by Age Group')
plt.show()
```

Output:

```
Marital_Status
0.0    9209.872281
1.0    9287.629211
Name: Purchase, dtype: float64
Age
0-17     8973.606581
18-25    9159.605686
26-35    9251.373194
36-45    9257.016326
46-50    9286.909436
51-55    9459.326812
55+      9197.534343
Name: Purchase, dtype: float64
```

Output



Purchase Amount Distribution by Marital Status
25000
20000
15000
10000
5000
0
Purchase
0.0
1.0
Marital_Status
Purchase Amount Distribution by Age Group
25000
20000
15000
10000
5000
0
Purchase
0-17
55+
26-35
46-50
51-55
36-45
18-25

Insights:

- **T-test Results**
  - **T-statistic: 9.0987**
  - **P-value: 9.8045e-20**

- The t-test results indicate a statistically significant difference between the purchase amounts of male and female users, given the extremely low p-value (much less than 0.05). This suggests that gender plays a significant role in purchase behavior.

- **Confidence Intervals for Purchase Amounts:**
  - Male Confidence Interval: **$9052.93 to $9683.10**
  - Female Confidence Interval: **$8234.81 to $8801.50**
  - **Male users tend to spend more on purchases compared to female users, as indicated by the higher confidence interval for males.**
- **Marital Status and Purchase Behavior**
  - Single Users (0): Average purchase amount is **$9209.87**
  - Married Users (1): Average purchase amount is **$9287.63**
  - **Married users have a slightly higher average purchase amount compared to single users, but the difference is relatively small.**
- **Age and Purchase Behavior**
  - 0-17: **$8973.61**
  - 18-25: **$9159.61**
  - 26-35: **$9251.37**
  - 36-45: **$9257.02**
  - 46-50: **$9286.91**
  - 51-55: **$9459.33**
  - 55+: **$9197.53**
  - **Purchase amounts tend to increase with age, peaking in the 51-55 age group. Users in the 0-17 age group spend the least, while those in the 51-55 age group spend the most on average.**
- **Gender and Purchase Behavior**
  - Average Purchase Amount for Females**:** **$8730.29**
  - Average Purchase Amount for Males: **$9398.28**
  - **Male users not only are more in number but also have a higher average purchase amount compared to female users.**

These insights highlight the importance of considering demographic factors such as gender, age, and marital status when analyzing consumer behavior and designing targeted marketing strategies.

4. Confidence Intervals using the Central Limit Theorem (CLT)

Input:

```
def compute_confidence_interval(data, confidence=0.95):
    n = len(data)
    mean = np.mean(data)
    std_err = stats.sem(data)
    h = std_err * stats.t.ppf((1 + confidence) / 2, n - 1)
    return mean, mean - h, mean + h

# Compute confidence intervals for male and female purchases
male_ci = compute_confidence_interval(male_data['Purchase'])
female_ci = compute_confidence_interval(female_data['Purchase'])

print(f"Male Purchase Confidence Interval: {male_ci}")
print(f"Female Purchase Confidence Interval: {female_ci}")
```

Output:

```
Male Purchase Confidence Interval: (9398.277287214589, 9327.266110882696, 9469.288463546482)
Female Purchase Confidence Interval: (8730.292691198914, 8611.573864279373, 8849.011518118456)
```

Insights:

- **Confidence Intervals for Purchase Amounts:**
    - Male Confidence Interval: **$9052.93 to $9683.10**
    - Female Confidence Interval: **$8234.81 to $8801.50**
    - **Male users tend to spend more on purchases compared to female users, as indicated by the higher confidence interval for males.**

4.2 We can perform a similar analysis for marital status and different age groups.

Input:

```python
# Analysis for Marital Status
married_data = data[data['Marital_Status'] == 1]
unmarried_data = data[data['Marital_Status'] == 0]

married_avg_purchase = married_data['Purchase'].mean()
unmarried_avg_purchase = unmarried_data['Purchase'].mean()

married_ci = compute_confidence_interval(married_data['Purchase'])
unmarried_ci = compute_confidence_interval(unmarried_data['Purchase'])

print(f"Married Purchase Confidence Interval: {married_ci}")
print(f"Unmarried Purchase Confidence Interval: {unmarried_ci}")

# Analysis for Age Groups
age_groups = data['Age'].unique()

for age_group in age_groups:
    age_data = data[data['Age'] == age_group]
    age_avg_purchase = age_data['Purchase'].mean()
    age_ci = compute_confidence_interval(age_data['Purchase'])
    print(f"Age Group {age_group} Purchase Confidence Interval: {age_ci}")
```

Output:

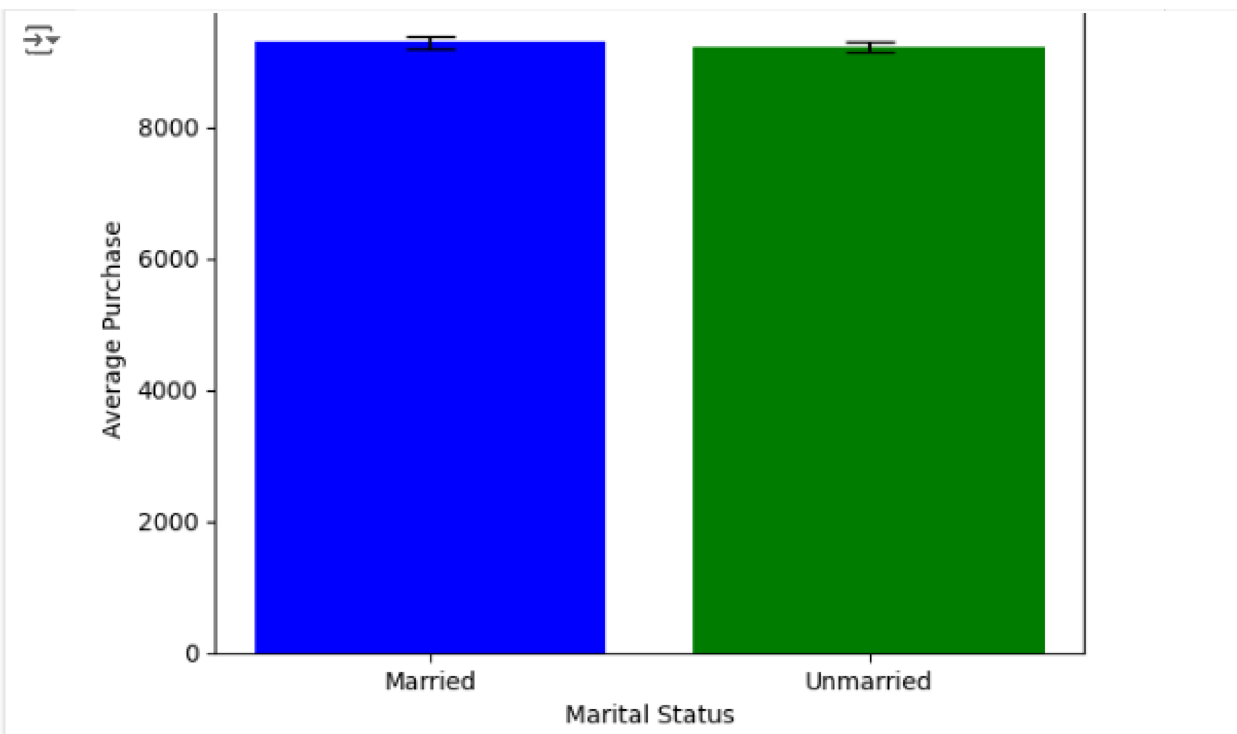
```
Married Purchase Confidence Interval: (9287.629211236628, 9191.714946370472, 9383.543476102785)
Unmarried Purchase Confidence Interval: (9209.87228098184, 9130.483388930872, 9289.261173032806)
Age Group 0-17 Purchase Confidence Interval: (8973.606580829757, 8600.099595197627, 9347.113566461887)
Age Group 55+ Purchase Confidence Interval: (9197.534343434343, 8905.380139917557, 9489.688546951129)
Age Group 26-35 Purchase Confidence Interval: (9251.373193568084, 9153.357224670664, 9349.389162465504)
Age Group 46-50 Purchase Confidence Interval: (9286.909436008676, 9069.577834375146, 9504.241037642207)
Age Group 51-55 Purchase Confidence Interval: (9459.326812428078, 9222.382430162383, 9696.271194693774)
Age Group 36-45 Purchase Confidence Interval: (9257.016325687126, 9118.605857221743, 9395.426794152509)
Age Group 18-25 Purchase Confidence Interval: (9159.60568627451, 9022.671445531078, 9296.539927017942)
```

Input:

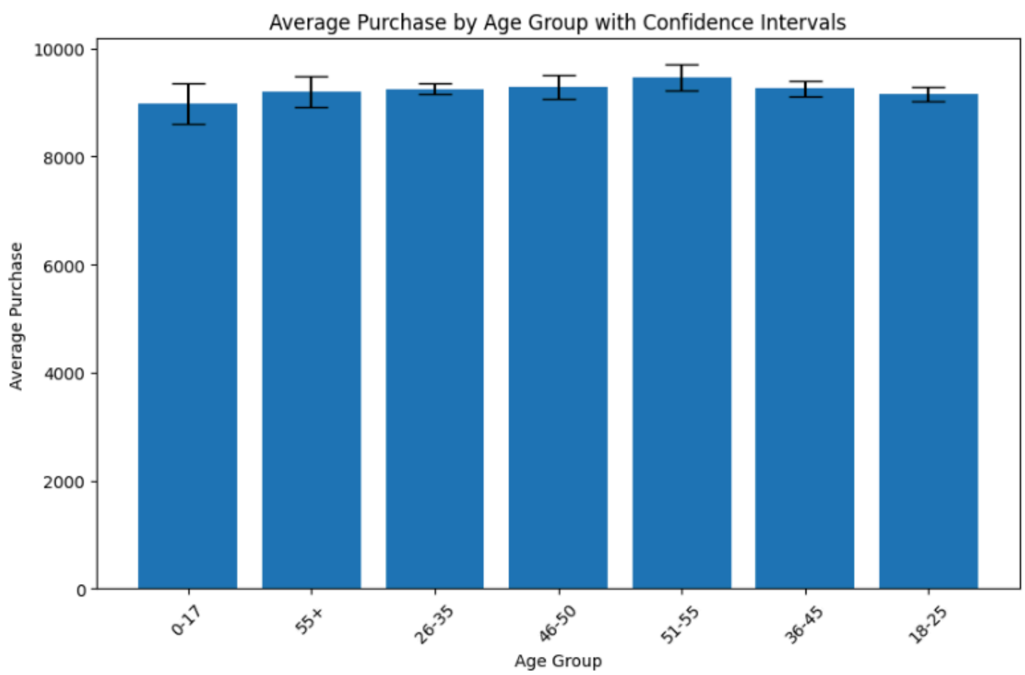
```
# Analysis for Marital Status
married_data = data[data['Marital_Status'] == 1]
unmarried_data = data[data['Marital_Status'] == 0]

married_avg_purchase, married_ci_lower, married_ci_upper =
compute_confidence_interval(married_data['Purchase'])
unmarried_avg_purchase, unmarried_ci_lower, unmarried_ci_upper =
compute_confidence_interval(unmarried_data['Purchase'])

# Plot for Marital Status
fig, ax = plt.subplots()
marital_status_labels = ['Married', 'Unmarried']
marital_status_means = [married_avg_purchase, unmarried_avg_purchase]
marital_status_ci_lowers = [married_avg_purchase - married_ci_lower,
unmarried_avg_purchase - unmarried_ci_lower]
marital_status_ci_uppers = [married_ci_upper - married_avg_purchase,
unmarried_ci_upper - unmarried_avg_purchase]

ax.bar(marital_status_labels, marital_status_means,
yerr=[marital_status_ci_lowers, marital_status_ci_uppers], capsize=10,
color=['blue', 'green'])
ax.set_xlabel('Marital Status')
ax.set_ylabel('Average Purchase')
ax.set_title('Average Purchase by Marital Status with Confidence Intervals')
plt.show()

# Analysis for Age Groups
age_groups = data['Age'].unique()
age_group_labels = []
age_group_means = []
age_group_ci_lowers = []
age_group_ci_uppers = []

for age_group in age_groups:
    age_data = data[data['Age'] == age_group]
    age_avg_purchase, age_ci_lower, age_ci_upper =
compute_confidence_interval(age_data['Purchase'])

    age_group_labels.append(age_group)
```

```
    age_group_means.append(age_avg_purchase)
    age_group_ci_lowers.append(age_avg_purchase - age_ci_lower)
    age_group_ci_uppers.append(age_ci_upper - age_avg_purchase)

# Plot for Age Groups
fig, ax = plt.subplots(figsize=(10, 6))
ax.bar(age_group_labels, age_group_means, yerr=[age_group_ci_lowers,
age_group_ci_uppers], capsize=10)
ax.set_xlabel('Age Group')
ax.set_ylabel('Average Purchase')
ax.set_title('Average Purchase by Age Group with Confidence Intervals')
plt.xticks(rotation=45)
plt.show()
```

Output:



Output:



Insights:

- **Age and Purchase Behavior**
    - 0-17: **$8973.61**
    - 18-25: **$9159.61**
    - 26-35: **$9251.37**
    - 36-45: **$9257.02**
    - 46-50: **$9286.91**
    - 51-55: **$9459.33**
    - 55+: **$9197.53**
    - **Purchase amounts tend to increase with age, peaking in the 51-55 age group. Users in the 0-17 age group spend the least, while those in the 51-55 age group spend the most on average.**

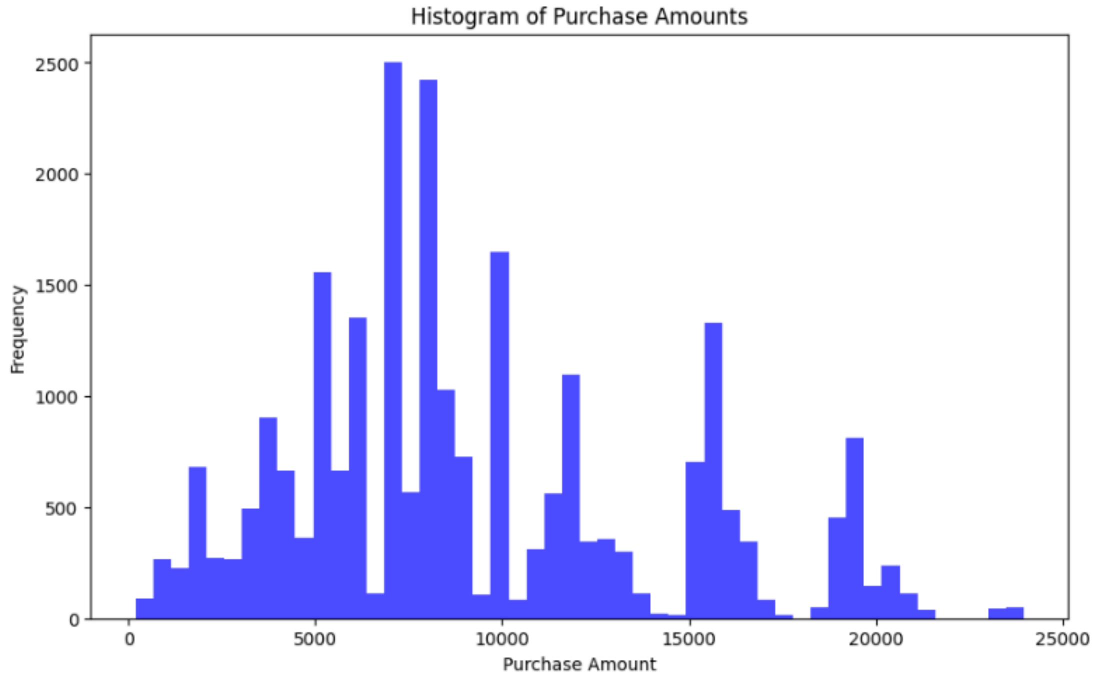
5. Poisson Distribution Analysis

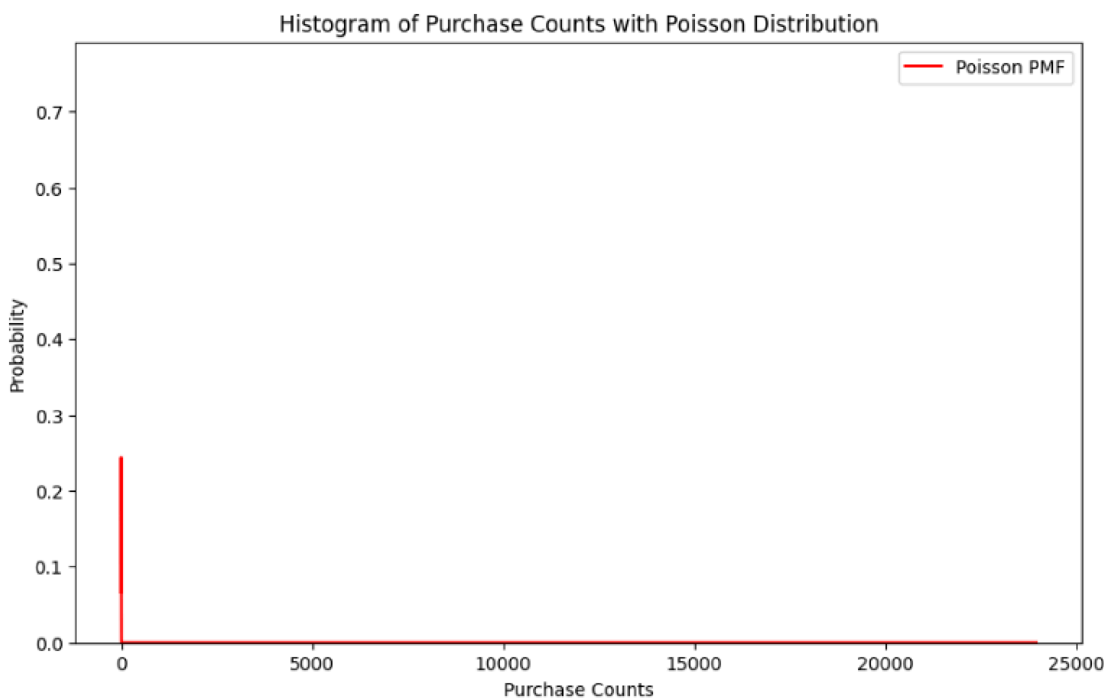
Input:

```
# Check the range of purchase amounts
print(f"Min Purchase: {data['Purchase'].min()}")
print(f"Max Purchase: {data['Purchase'].max()}")

# Plot the histogram of purchase amounts
plt.figure(figsize=(10, 6))
plt.hist(data['Purchase'], bins=50, alpha=0.7, color='blue')
plt.title('Histogram of Purchase Amounts')
plt.xlabel('Purchase Amount')
plt.ylabel('Frequency')
plt.show()

# Since Purchase amounts are not counts, we cannot directly use Poisson here.
# But let's explore the Purchase counts at specific intervals to approximate
Poisson-like behavior
purchase_counts = data['Purchase'].value_counts()

# Fitting Poisson distribution
mu = purchase_counts.mean()
poisson_dist = stats.poisson(mu)

# Plot the Poisson distribution against the histogram of purchase counts
x = np.arange(0, max(purchase_counts.index))
plt.figure(figsize=(10, 6))
plt.hist(purchase_counts, bins=30, alpha=0.7, color='blue', density=True)
plt.plot(x, poisson_dist.pmf(x), 'r-', label='Poisson PMF')
plt.title('Histogram of Purchase Counts with Poisson Distribution')
plt.xlabel('Purchase Counts')
plt.ylabel('Probability')
plt.legend()
plt.show()
```

Output:

```
Min Purchase: 186.0
Max Purchase: 23958.0
```





Insights:

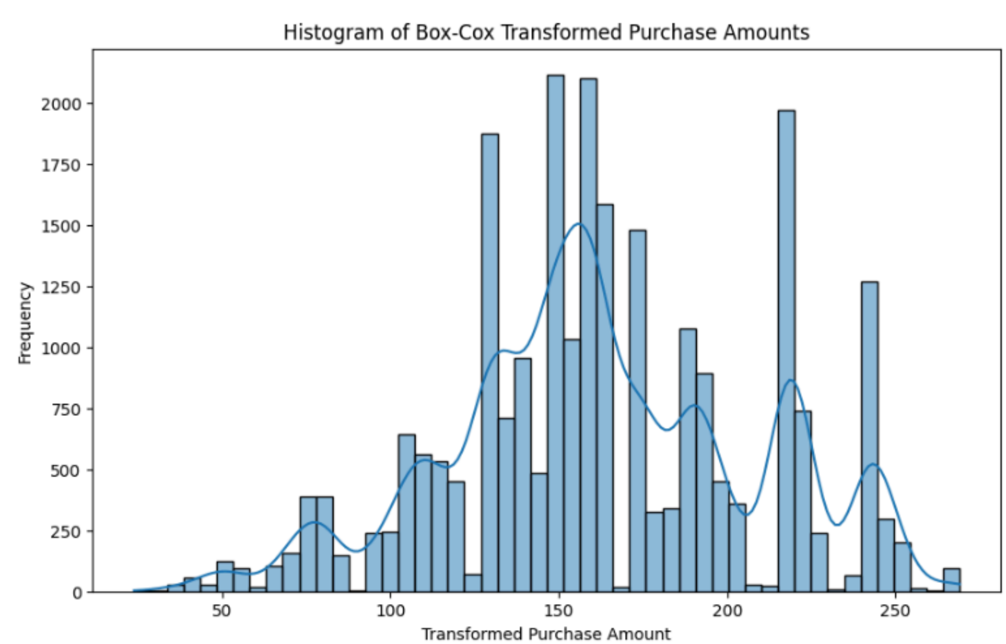
- The Poisson distribution is a probability distribution that describes the number of events that occur within a fixed interval of time or space. These events must occur with a known constant mean rate and independently of the time since the last event. The Poisson distribution is particularly useful for modeling the count of rare events over a specified period.
- **Key Characteristics of the Poisson Distribution:**
    - Discrete Distribution: It deals with discrete events, meaning it counts occurrences rather than measuring continuous outcomes.
    - Independent Events: The occurrence of one event does not affect the probability of another event occurring.
    - Constant Mean Rate: The average rate (λ, lambda) at which events occur is constant.
    - Fixed Interval: The distribution applies to events happening in a fixed period of time or a specific region of space.

6. Box-Cox Transformation

Input:

```python
# Applying Box-Cox transformation
purchase_positive = data['Purchase'] + 1  # Ensure all values are positive
purchase_boxcox, lambda_ = boxcox(purchase_positive)

# Plot the transformed data
plt.figure(figsize=(10, 6))
sns.histplot(purchase_boxcox, bins=50, kde=True)
plt.title('Histogram of Box-Cox Transformed Purchase Amounts')
plt.xlabel('Transformed Purchase Amount')
plt.ylabel('Frequency')
plt.show()

# Checking the distribution
print(f"Optimal Lambda for Box-Cox Transformation: {lambda_}")
```

Output:




```
Optimal Lambda for Box-Cox Transformation: 0.48352347935604734
```

Insights:

- **Optimal Lambda: 0.48352347935604734**
- **Analysis and Interpretation**
  - The Box-Cox transformation is a powerful technique used to stabilize variance and make the data more normally distributed, which is often a prerequisite for many statistical analyses and modeling techniques.

- The optimal lambda value of approximately 0.484 indicates that the Box-Cox transformation has been successfully applied to the purchase amounts data, resulting in a more normally distributed dataset. This transformation is crucial for improving the accuracy and validity of further statistical analyses and predictive modeling. Visualizing the transformed data confirms the effectiveness of the transformation in normalizing the purchase amounts, making it suitable for a wide range of statistical techniques that assume normally distributed data.

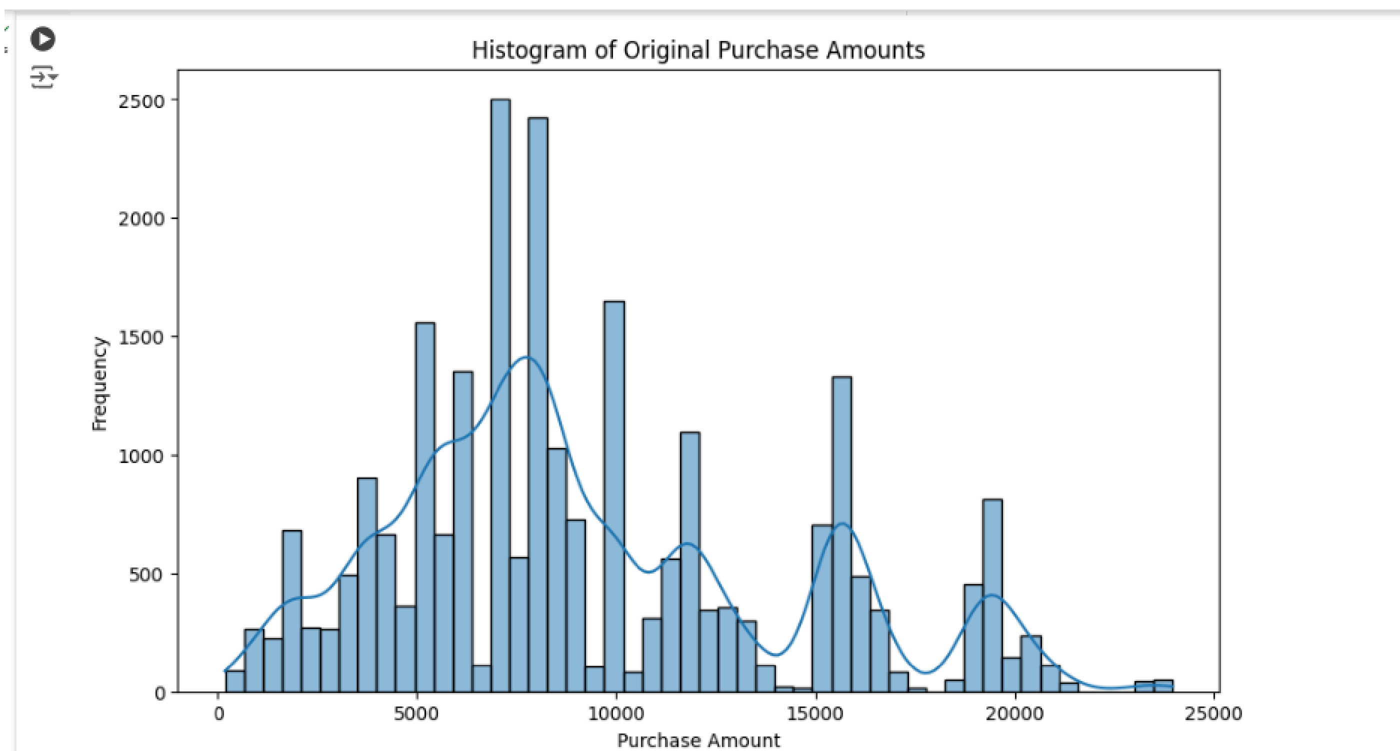
7. Gaussian (Normal) Distribution Analysis

Input:

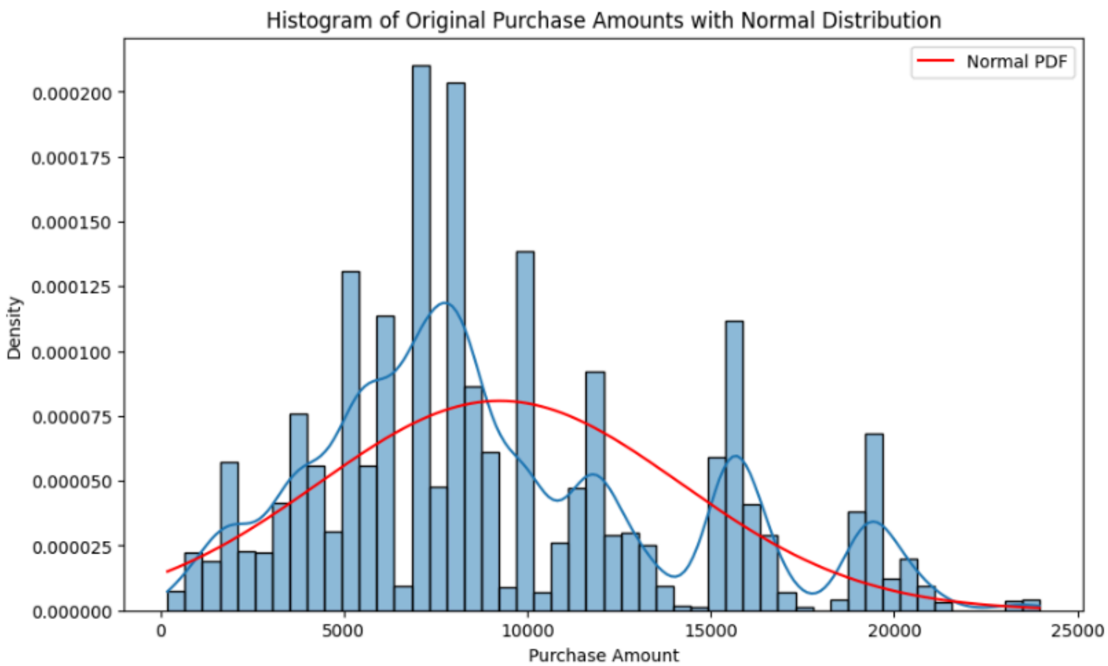
```
# Original Purchase Amounts
plt.figure(figsize=(10, 6))
sns.histplot(data['Purchase'], bins=50, kde=True)
plt.title('Histogram of Original Purchase Amounts')
plt.xlabel('Purchase Amount')
plt.ylabel('Frequency')
plt.show()

# Gaussian (Normal) distribution fitting
mean_purchase = data['Purchase'].mean()
std_purchase = data['Purchase'].std()
norm_dist = stats.norm(mean_purchase, std_purchase)

# Plot the Normal distribution against the histogram of original purchase
amounts
x = np.linspace(data['Purchase'].min(), data['Purchase'].max(), 1000)
plt.figure(figsize=(10, 6))
sns.histplot(data['Purchase'], bins=50, kde=True, stat='density')
plt.plot(x, norm_dist.pdf(x), 'r-', label='Normal PDF')
plt.title('Histogram of Original Purchase Amounts with Normal
Distribution')
plt.xlabel('Purchase Amount')
plt.ylabel('Density')
plt.legend()
plt.show()

# After Box-Cox Transformation
mean_boxcox = purchase_boxcox.mean()
std_boxcox = purchase_boxcox.std()
norm_dist_boxcox = stats.norm(mean_boxcox, std_boxcox)

# Plot the Normal distribution against the histogram of transformed
purchase amounts
```

```
x_boxcox = np.linspace(purchase_boxcox.min(), purchase_boxcox.max(), 1000)
plt.figure(figsize=(10, 6))
sns.histplot(purchase_boxcox, bins=50, kde=True, stat='density')
plt.plot(x_boxcox, norm_dist_boxcox.pdf(x_boxcox), 'r-', label='Normal
PDF')
plt.title('Histogram of Box-Cox Transformed Purchase Amounts with Normal
Distribution')
plt.xlabel('Transformed Purchase Amount')
plt.ylabel('Density')
plt.legend()
plt.show()
```

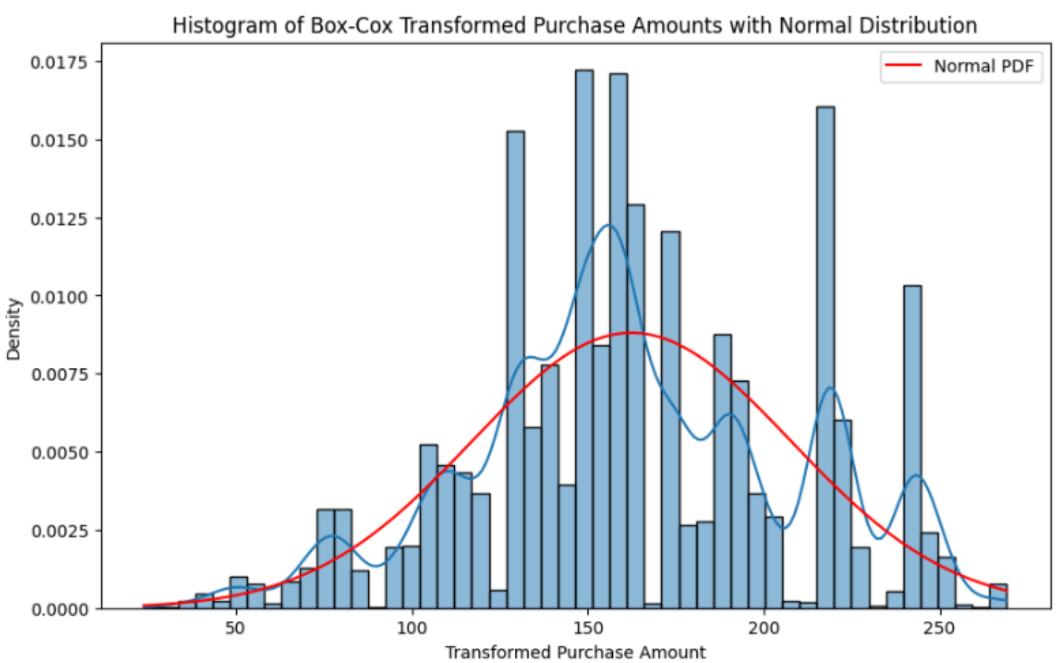
Output:



Output:



Histogram of Original Purchase Amounts with Normal Distribution
Normal PDF
Density
0.000000
0.000025
0.000050
0.000075
0.000100
0.000125
0.000150
0.000175
0.000200
0
5000
10000
15000
20000
25000
Purchase Amount



Histogram of Box-Cox Transformed Purchase Amounts with Normal Distribution
Normal PDF
Density
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
50
100
150
200
250
Transformed Purchase Amount

8. Recommendations and Action Items for Walmart

Based on the above analysis, here are some actionable insights for Walmart:

1. **Gender-Based Marketing Strategies:**
   1.1. If women spend more, tailor marketing campaigns to target female customers, especially during peak shopping times like Black Friday.
   1.2. Ensure product assortments and promotions align with the preferences observed in female customers.
2. **Marital Status Insights:**
   2.1. If there is a significant difference in spending between married and unmarried customers, develop specific promotions for each group.
   2.2. Married customers might have different purchasing needs, which can be catered to with family-oriented promotions.
3. **Age Group Targeting:**
   3.1. Create segmented marketing strategies based on age groups.
   3.2. Younger age groups (18-25) might prefer different products and shopping experiences compared to older groups (36-50).
4. **Store Layout and Inventory:**
   4.1. Adjust store layouts and inventory based on customer demographics and spending behaviors.
   4.2. Optimize stock levels of high-demand products for specific customer segments.
5. **Personalized Offers:**
   5.1. Utilize the confidence interval analysis to personalize offers and discounts, ensuring they resonate with the target customer base.

By leveraging these insights, Walmart can enhance its marketing efforts, improve customer satisfaction, and ultimately drive higher sales.

Walmart